





MONTAJE BÁSICO DE LINUX CLUSTERS
UNA GUÍA PRÁCTICA

Última Actualización: 11 de julio de 2008
Elaborado por: Jorge Zuluaga

Presentación

Esta guía práctica describe la manera como se prepara, monta, instala, configura y prueba un Cluster Computacional usando nodos que corren el sistema operativo Linux. El propósito fundamental de este manual es el de servir como documento de referencia para un ejercicio práctico de instalación de lo que en lo sucesivo llamaremos "Linux Cluster", que puede ser utilizado o bien con propósitos didácticos y de entrenamiento o para el despliegue de un Cluster de Producción.

El manual hace parte de la colección "Guías Prácticas de Computación" desarrollado inicialmente por el Prof. Jorge Zuluaga con el apoyo del Grupo de Física y Astrofísica Computacional del Instituto de Física de la Universidad de Antioquia y complementado desde Mayo de 2007 a través del apoyo del Centro Regional de Simulación y Cálculo Avanzado (CRESCA) del que es socio fundador la Universidad de Antioquia.

La colección de Guías Prácticas esta actualmente formada por esta Guía y una Guía Práctica de Linux Clustering con Rocks que es la distribución utilizada en el CRESCA. Los manuales pueden descargarse libremente del sitio del Grupo de Física y Astrofísica Computacional http://urania.udea.edu.co/facom. Especialmente importantes son las actualizaciones que de forma no regular (dependiendo de las necesidades que surgen en el medio) se realizan de esas mismas guías.

Esperamos que este documento sea de utilidad para todos aquellos que o bien necesitan utilizar herramientas como los Clusters Computacionales para la realización de tareas de computación de Alto Rendimiento en Ciencias e Ingeniería o para quienes se valen de este conocimiento para formar estudiantes de Ingeniería o capacitar a personal especializado en el área.

Jorge Zuluaga
Medellín, 2008

Convenciones

La guía contiene una colección de comando útiles, tips de uso, scripts y comentarios generales sobre un conjunto amplio de tópicos relacionados con el uso y administración de Clusters Rocks. Se asume en esta guía las siguientes convenciones tipográficas:

Convención	**Explicación**	**Ejemplo**
Comandos #,#: # comando	*Comandos de linux. Todos los comando vienen numerados de acuerdo a la sesión en la que aparecen. El nombre del comando se indica seguido del símbolo del sistema: '#' cuando el comando debe ser ejecutado como administrador y '$' cuando el comando debe ser ejecutado como usuario. '%' será utilizado también cuando el comando pueda ser ejecutado con permisos de root o de usuario.*	Comando 1,20: # ls /var/log/message*
Salida: `salida en pantalla`	*Cuando se muestre la salida de un comando esta se presenta en tipo* `courier` *pequeño.*	Salida: `/var/log/messages` `/var/log/messages.2` `/var/log/messages.4` `/var/log/messages.1` `/var/log/messages.3`
Archivo: `...` `#Configuration file` `FILE=configuration` `SERVER=localhost` `...`	*El contenido de un archivo se presenta en tipo courier con una sangría respecto al resto del documento y con una línea vertical que delimita el contenido del archivo.* *Cuando el documento no esta completo se colocan puntos suspensivos arriba o abajo indicando la presencia de más líneas*	./run.sh: `#!/bin/bash` `#Script para SGE` `#-S /bin/bash` `#-j y` `dir=/home/fulano/run1` `cd $dir` `./program.out`
Descargue: archivo	*Cuando en una situación dada se requiera un archivo o un paquete especial se referirá al lector al sitio del Grupo FACom donde podrá encontrar el archivo respectivo.* *El sitio es:* *http://urania.udea.edu.co/facom* *Y los archivos se encuentran en el enlace "documentación"*	Descargue: run.sh

Parte 1. Preparación

Las condiciones básicas para crear un Cluster Linux son bastante simples de conseguir e incluso se cumplen en muchos entornos académicos y corporativos. En esta parte describimos como se prepara el material de Hardware y Software necesario para el despliegue de un cluster Linux. Una característica muy importante del tipo de Cluster cuya instalación se describe aquí es que es un Cluster que se construye desde “cero”, es decir, partiendo de la instalación básica del sistema operativo para a continuación procedes con la instalación y configuración de todo el software necesario para el computo y el almacenamiento distribuido. Este tipo de clusters se contraponen a quellos clusters cuya instalación y configuración se apoyan de piezas de software específicas y complejas, a veces de instalación muy sencilla y que incluyen como componentes las herramientas que describimos aquí. Este último tipo de herramientas permiten normalmente desplegar un cluster completamente operativo en muy poco tiempo. Herramientas como OSCAR, Rocks, OpenMosix son herramientas de este tipo. El caso de Rocks se documenta en otra de las Guías Prácticas de esta colección.
El montaje, instalación y configuración de un Cluster básico ofrece ventajas didácticas y técnicas que lo hacen muy interesante para el propósito de esta Guía.

1.1. Condiciones iniciales

Para el montaje de un cluster linux como el descrito en este documento se deben cumplir los siguientes requerimientos mínimos de hardware y software:

Hardware

1. Disposición de 2 o más computadores, con prestaciones mínimas: >512 MB RAM, >10 GB DD, entre 1 y 2 tarjetas de red alámbrica para cada equipo. Por el tipo de instalación que se describe aquí se requiere también contar con los periféricos básicos en cada máquina: monitor, teclado, mouse. También es posible utilizar un switch KVM para simplificar la entrada y salida.

2. Los equipos deben estar interconectados a través de una red. Para ello se requieren los medios físicos de conexión (cableado) y equipos para el intercambio de paquetes (hub, switch, enrutador).

NOTA: Todo lo anterior puede reemplazarse por medios virtuales en caso de que se cuente con un equipo con buenas prestaciones y se utilicen máquinas virtuales para crear las componentes del cluster. Los detalles del proceso en este caso, a diferencia del proceso de instalación, son los mismos.

Software

3. Medios de instalación de una distribución convencional de Linux (Linux Fedora, Linux CentOS, Linux Redhat, Debian, etc.) Los comandos y tareas de configuración mencionados en esta guía suponen que se cuenta con un sistema compatible con distribuciones Redhat (CentOS, Fedora, Redhat EL, etc.) Otras distribuciones no compatibles también pueden utilizarse con algunos cambios en los comandos administrativos. Es recomendable haber probado todos los medios antes de avanzar con la instalación.

4. Paquetes básicos que deben ser incluidos en la instalación (y que en ocasiones o en ciertas configuraciones pre establecidas no se incluyen):

    1. openssh server
    2. autofs
    3. nfs
    4. ypserv, ypbind, yp-tools

    Si la instalación del sistema se ha hecho previa a la realización del ejercicio propuesto en esta guía es posible verificar la presencia de los paquetes usando la opción query del comando rpm:

Comandos 1:

```
# rpm -qa | grep <paquete>

Ejemplo:
# rpm -qa | grep openssh-server
```

Si alguno de los paquetes no esta instalado debe instalarse satisfaciendo todas las dependencias exigidas.

1.2. Configuración de la red

El acceso a la red que comparten las máquinas en el cluster es la condición fundamental para garantizar su constitución como plataforma distribuida. Es necesario como condición básica configurar inicialmente la red y otros aspectos relacionados con la interconectividad entre las máquinas.

Linux cuenta con multiples aplicativos que permiten la configuración de la red de forma amigable usando para ello interfaces gráficas (tanto en la consola de texto como en los gestores de escritorio). Describiremos aquí para facilitar eventualmente la automatización de los procesos de preparación de la red la configuración usando directamente los archivos de texto plano que usa linux para tal fin.

1.2.1. Configuración de los parámetros básicos

Para configurar los parámetros básicos de la red (IP, Netmask, Gateway, nombre de la máquina, dominio) deben editarse los archivos **/etc/sysconfig/network** y **/etc/sysconfig/network-scripts/ifcfg-eth0**. Suponiendo como valores válidos los parámetros descritos a continuación:

IP: 10.X.X.X
Netmask: 255.0.0.0
Gateway: 10.1.1.1
Broadcast: 10.0.0.0
DNS primario: 10.1.1.254
Nombre: server, nodoX
Dominio del cluster: cluster

Los archivos mencionados deben contener las siguientes entradas:

Archivo /etc/sysconfig/network:

```
NETWORKING=yes
NETWORKING_IPV6=no
HOSTNAME=nodoX.cluster
GATEWAY=10.1.1.1
NISDOMAIN=cluster
```

Archivo /etc/sysconfig/network-scripts/ifcfg-eth0:

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=static
IPADDR=10.1.1.X
NETMASK=255.0.0.0
GATEWAY=10.1.1.1
```

NOTA: Si los equipos pertenecen a una red con DHCP los archivos anteriores deben dejarse tal y como fueron configurados en la instalación. Solamente el NISDOMAIN puede tener un valor arbitrario independiente del dominio real al que pertenezca la máquina.

Una vez configurada la red de la forma indicada arriba es posible arrancar (start) la interface de red respectiva:

Comandos 2:

ifup eth0

```
eth0      Link encap:Ethernet  HWaddr 08:00:27:60:45:85
          inet addr:10.1.1.1  Bcast:10.255.255.255  Mask:255.0.0.0
          inet6 addr: fe80::a00:27ff:fe60:4585/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:157763 errors:53 dropped:0 overruns:0 frame:0
          TX packets:395612 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:46637638 (44.4 MiB)  TX bytes:566159863 (539.9 MiB)
          Interrupt:11 Base address:0xc020

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:120203 errors:0 dropped:0 overruns:0 frame:0
          TX packets:120203 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:25196071 (24.0 MiB)  TX bytes:25196071 (24.0 MiB)
```

NOTA: Si los equipos pertenecen a una red con DHCP los archivos anteriores deben dejarse tal y como fueron configurados en la instalación. Solamente el NISDOMAIN puede tener un valor arbitrario independiente del dominio real al que pertenezca la máquina.

1.2.2. Configuración de otros parámetros de red

Otros parámetros de la red pueden configurarse para abreviar las comunicaciones entre las máquinas del cluster. En particular es posible configurar el archivo /etc/hosts para facilitar la conversión de nombres de máquinas en la red del clusters a IP y viceversa y sin requerir una

compleja consulta a un servidor de nombres.

Archivo /etc/hosts:

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1       localhost.localdomain   localhost

10.1.1.1       server.cluster  server
10.1.1.2       nodo.cluster    nodo
```

Es necesario recordar que la primera línea no debe bajo ninguna circunstancia modificarse porque como se explica en el archivo muchos programas dependen de ella.

También es posible que sea necesario configurar el acceso a un servidor de nombres externo que permita hacer traducción de nombres a IPs de máquinas en una red externa al cluster. Para hacerlo basta con editar el archivo /etc/resolv.conf:

Y que el servidor de nombres es 192.168.1.34, la configuración de los archivos mencionados se leería:

Archivo /etc/resolv.conf:

```
search cluster
search domain.institution
nameserver 10.1.1.254
```

Una vez fijados los anteriores archivos de configuración es recomendable reinciar completamente las máquinas con el fin de garantizar que todos los programas en ejecución o que se habilitan durante el arranque reciban los valores de configuración correctos.

1.3. Pruebas básicas de la red

Si la configuración de red funciona apropiadamente es posible realizar algunas pruebas básicas de conectividad para garantizar que los nodos puedan intercambiar información entre sí. Para ello se recomienda realizar el siguiente conjunto de pruebas.

1.3.1. Prueba con ping

Se puede probar la conectividad básica a la red haciendo una prueba de ping al Gateway (si existe) y a otras máquinas en la red interna del cluster:

Comandos 3:

```
# ping -c 3 10.1.1.1
# ping -c 3 10.1.1.2
# ping -c 3 server
# ping -c 3 nodo8.cluster
```

Las pruebas con los FQHN (Fully Qualified Hostname) y con los aliases definidos en /etc/hosts son fundamentales para garantizar que los mecanismos definidos en este archivo estén funcionando correctamente.

1.3.2. Medida de rendimiento usando netperf

Es posible antes de empezar a usar el cluster realizar algunas medidas del rendimiento de la red usando para ello un paquete como netperf. Para ello pueden utilizarse algunas herramientas libres disponibles en la comunidad de linux. Una herramienta liviana y sencilla de configurar, usar y entender es netperf.

Naturalmente antes de usar netperf o sus componentes es necesario instalar el paquete respectivo (ver consecución

Para usar netperf se debe primero arrancar el servidor netserver en cada una de las máquinas del cluster:

Comandos 4:

```
# ./netserver
```

Una vez iniciados los servidores en cada máquina desde cualquier punto en la red se puede lanzar un comando de prueba:

Comandos 5:

```
# netperf -l 30 -H 10.1.1.2 -t TCP_STREAM

TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to 10.1.1.2 port 0 AF_INET
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    30.00    5556.15
```

Donde 30 es un indicador de la longitud del paquete (tiempo de transferencia en segundos) y en lugar de usar el protocolo TCP (doble vía) se puede usar el protocolo UDP (una sola vía, UDP_STREAM).

1.3.3. Pruebas de conectividad con ssh

Para la autenticación de usuario y el manejo de consolas remotas en las máquinas en el cluster se usará el protocolo securo openssh. Antes de continuar con otras configuraciones es necesario garantizar que el servicio este activo en todas las máquinas en el cluster.

NOTA:

Como referencia para lo sucesivo los servicios en linux se inician, detienen y verifican usando una de dos maneras posibles:

1. Comando 'service' (no disponible en todas las distribuciones).

   - Verificación del servicio: service <servicio> status

Comandos 6:
Ejemplo:
service sshd status

- Inicio de servicios: service <servicio> start
- Parada de servicio: service <servicio> stop

2. Ejecución directa del script del demonio asociado al servicio:
Todo servicio en Linux involucra el que se denomina un demonio, un programa que esta activo en backgroun escuchando los requerimientos que se hacen al servicio y ejecutando las acciones que se solicitan. Los demonios de la mayoría de los servicios de linux se ubican en el directorio /etc/rc.d/init.d.

- Verificación del servicio: /etc/rc.d/init.d/<demonio> status

Comandos 6:
/etc/init.d/sshd status

- Inicio de servicios: /etc/init.d/<demonio> start

- Parada de servicio: /etc/init.d/<demonio> stop

Arrancar un servicio que se encontraba parado garantiza que este el mismo servicio se preste mientras la máquina este arriba. Para activar el servicio automáticamente en el momento del arranque se debe configurar apropiadamente el sistema de arranque de linux. Esto se realiza usando el comando 'chkconfig'.

Comandos 7:
Para verificar si un servicio se encuentra habilitado al momento del arranque:
chkconfig --list sshd
```
sshd            0:off      1:off  2:on   3:on   4:on   5:on   6:off
```

Para habilitar servicio al momento del arranque:
chkconfig --levels 345 sshd on

Para deshabilitar servicio al momento del arranque:
chkconfig --levels 345 sshd off

Continuando con el orden de ideas, para garantizar que la conectividad con openssh funcione es necesario verificar que el servicio sshd este corriendo y de lo contrario ponerlo a funcionar y habilitarlo al arranque.

Una vez funcionando se puede verificar su funcionamiento abriendo una consola remota con el comando 'ssh' así:

Comandos 8:
ssh root@10.1.1.2
```
The authenticity of host '10.1.1.2' can't be established.
RSA key fingerprint is ca:50:6f:3e:12:a2:29:86:09:b2:89:f1:e6:1b:74:06.
```

```
Are you sure you want to continue connecting (yes/no)? Yes
Warning: Permanently added '10.1.1.2' (RSA) to the list of known hosts.
root@10.1.1.2's password:<PASSWORD>
```

Una vez llegamos a este punto podemos afirmar que las condiciones inciales necesarios para la configuración de los servicios de computo y almacenamiento distribuido están garantizadas.

Parte 2. Configuración de los servicios básicos

2.1. Configuración de los sistema de información

Una de las características importantes de un Cluster de Computadores es que la información de configuración de algunos servicios críticos en todas sus componentes debe estar adecuadamente configurada. Así por ejemplo la lista de los usuarios reconocidos en el sistema y sus contraseñas deben ser compartidas por todas las componentes del sistema. Para ello se han desarrollado complejos y muy robustos sistemas de información que garantizan que la información de configuración presente en una máquina (el frontend) sea conocida por todas las demás en el cluster (nodos).

En esta guía utilizaremos el sistema NIS por su versatilidad y fácil manejo.

El NIS (Network Information System) es un sistema que provee información de configuración para máquinas conectadas en red en un esquema de cliente-servidor, incluyendo la información de autenticación. En términos más prácticos un sistema del tipo NIS permite que la tabla de passwords y de grupos de usuarios de un servidor pueda ser usada por otras máquinas en una red permitiendo a un usuario autenticarse en esas máquinas como si lo estuviera haciendo en el servidor.

El NIS funciona con un esquema de cliente-servidor. El servidor provee los archivos de configuración que se quieren proveer (generalmente passwd, group, hosts entre otros) y mantiene actualizados a los clientes con esta información. Los clientes son máquinas que reciben inicialmente los archivos de configuración y son receptores de cualquier actualización que se haga de ellos en el servidor.

La configuración de un sistema con NIS se produce a nivel del servidor y del cliente y se explica a continuación.

NOTA: En lo sucesivo asumimos que la red en todos los equipos del sistema ha sido configurada correctamente y que se encuentra habilitado en todos esos mismos equipos el servicio de **portmap**. Para verificar que el servicio se encuentra activo y el demonio esta ejecutándose deben utilizarse los siguientes comandos:

Comandos 9:

```
# chkconfig --list portmap
# /etc/init.d/portmap status
```

Si el demonio esta detenido o el servicio deshabilitado es necesario activarlo. Los comandos para hacerlo son:

Comandos 10:

```
# chkconfig --level 345 portmap on
# /etc/init.d/portmap start
```

2.1.1. Configuración básica de NIS

El primer paso para configurar el servidor es el de definir el "dominio NIS" que atenderá. El dominio NIS es un dominio virtual al que se encontrarán suscritos los clientes que comparten con el servidor la información de configuración.

Para verificar a que dominio pertenece un equipo se ejecuta el comando:

Comandos 11:

```
# domainname
```

Si el dominio al que pertenece el equipo ha sido apropiadamente configurado en el archivo /etc/sysconfig/network el anterior comando devolverá el nombre fijado con la variable NISDOMAIN. Si no se tiene fijado todavía un nombre de dominio NIS se lo puede fijar usando el comando:

Comandos 12:

```
# domainname <dominio>
```

Una vez asignado un nombre de dominio debe iniciarse el demonio del servidor ypserv y habilitar el servicio en tiempo de booteo:

Comandos 13:

```
# /etc/init.d/ypserv start
# chkconfig --level 345 ypserv on
```

Una vez activado el demonio se esta listo para preparar las bases de datos que contendrán la información de configuración que será compartida con los clientes. El comando para inicializar esas bases de datos es:

Comandos 14:

```
# /usr/lib/yp/ypinit -m
At this point, we have to construct a list of the hosts which will run NIS
servers.  Server.cluster is in the list of NIS server hosts.  Please continue to add
the names for the other hosts, one per line.  When you are done with the
list, type a <control D>.
      next host to add:  server.cluster
      next host to add:  <CTRL>+<D>
The current list of NIS servers looks like this:

sever.cluster

Is this correct?  [y/n: y]
We need a few minutes to build the databases...
Building /var/yp/facom/ypservers...
Running /var/yp/Makefile...
gmake[1]: Entering directory `/var/yp/facom'
```

```
Updating passwd.byname...
Updating passwd.byuid...
Updating group.byname...
Updating group.bygid...
Updating hosts.byname...
Updating hosts.byaddr...
Updating rpc.byname...
Updating rpc.bynumber...
Updating services.byname...
Updating services.byservicename...
Updating netid.byname...
Updating protocols.bynumber...
Updating protocols.byname...
Updating mail.aliases...
gmake[1]: Leaving directory `/var/yp/facom'

server.cluster has been set up as a NIS master server.

Now you can run ypinit -s server.cluster on all slave server.
```

Este comando crea en el directorio /var/yp un directorio con el nombre del dominio nis definido y que contiene una base de datos por cada archivo de configuración que se esta compartiendo (p.e. passwd.byname, group.byname, etc.)

Completado este procedimiento la máquina esta preparada para prestar el servicio NIS a otras máquinas en la red que así lo requieran

2.1.2. Configuración del cliente NIS en los nodos

La configuración del cliente se realiza en 3 sencillos pasos:

1. Se fija el dominio NIS al que se quiere vincular el cliente. Se utiliza para ello el comando domainname o el archivo de configuración /etc/sysconfig/network que usamos también para fijar el nombre del dominio en el caso del servidor.

2. Se fija el servidor del dominio NIS en el archivo de configuración /etc/yp.conf. El archivo debe verse así:

Archivo /etc/yp.conf:

```
# /etc/yp.conf - ypbind configuration file
# Valid entries are
domain cluster server server.cluster
#       Use server HOSTNAME for the domain NISDOMAIN.
#
#domain NISDOMAIN broadcast
#       Use  broadcast  on  the local net for domain NISDOMAIN
#
#ypserver HOSTNAME
#       Use server HOSTNAME for the  local  domain.  The
#       IP-address of server must be listed in /etc/hosts.
```

Aquí el 'server' debe escogerse con el nombre completo, el nombre abreviado o la IP del servidor yp que se escogió.

3. Se inicia el demonio ypbind:

Comandos 15:
```
# /etc/init.d/ypbind start
```

NOTA: La activación del demonio funcionará siempre y cuando se hayan ejecutado correctamente los pasos anteriores y el servidor NIS funcione apropiadamente.

Una vez iniciado el cliente NIS es necesario configurar el sistema para que la información de configuración sea leída del NIS y no de los archivos que residen en la máquina. Esta configuración se realiza modificando el archivo nsswitch.conf.

Allí se puede fijar con que prioridad son leidos los archivos de configuración, siendo files los archivos que se encuentran en el cliente y nis los que pertenecen al sistema NIS.

Por ejemplo para que los archivos de autenticación y de grupos de usuario se lean del NIS antes que de los archivos se deben fijar las siguientes líneas del archivo /etc/nsswitch.conf:

Archivo /etc/nsswitch.conf:
```
...
passwd:     nis files
shadow:     nis files
group:      nis files
hosts:      nis files dns
services:  nis files
...
```

Una vez configurado correctamente el cliente se procede a probar que la información de configuración este disponible. Esto se puede hacer ejecutando uno de los siguientes comandos:

Comandos 16:
```
# ypwhich
# ypcat passwd
```

La prueba más apropiada del funcionamiento del cliente de NIS es realizar una conexión al cliente con la información de autenticación del servidor. Para ello intente conectarse usando por ejemplo ssh en el cliente con el login y el password del servidor:

Comandos 17:
```
# ssh <login_servidor>@10.1.1.2
password: <password_servidor>
```

Si la autenticación es exitosa se abre un nuevo shell en el cliente pero el usuario no tendrá asignado un home directory sencillamente porque ese directorio se encuentra en el servidor y no en el cliente.

Para conseguir que el home directory sea accedido también en el cliente se usan los servicios NFS y autofs que se configuran a continuacion.

2.1.3. Adminsitración de NIS

Al cambiar los archivos de configuración en el servidor de NIS es siempre necesario enviar los archivos a todos los clientes suscritos al dominio. Por ejemplo cuando se cambia un password de un usuario en el servidor es necesario realizar esta operación inmediatamente despues. El comando para realizar esta operación es:

Comandos 18:

```
# make -C /var/yp
```

Cuando un usuario quiera cambiar su password puede utilizar el comando yppasswd. Para hacerlo deberá estar activado el demonio yppasswdd en el servidor NIS.

Un usuario autenticado en un cliente puede intentar modificar la información de autenticación. Usando comandos convencionales como passwd solo se modifican los archivos locales del cliente. Para modificar los archivos en el servidor se utiliza el comando yppasswd que funciona de la misma manera que el passwd solo que actualiza la tabla de shadow en el servidor y no en el cliente.

NOTA: para que esto tenga efecto es necesario habilitar el servicion yppasswdd en el servidor.

2.2. Configuración de los sistemas de archivos por red

Como vimos hacia el final de la sección anterior, uno de los problemas al intentar conectarnos en el cliente con una cuenta de usuario del servidor, si bien logramos autenticarnos la sesión se abre sobre un directorio que nada tiene que ver con el directorio “home” del usuario ...

2.2.1. El sistema NFS

NFS (Network Filesystem) es un sistema que permite accesar localmente un sistema de archivos que se encuentra en un dispositivo físico remoto. En términos practicos el NFS permite tener acceso inmediato a los archivos de otra máquina como si fueran archivos de una máquina local. NFS utiliza los protocolos RPC.

El primer paso para realizar una operación del tipo NFS es preparar la configuración en la máquina que exportará el sistema de archivos para definir quienes pueden acceder al sistema de archivos y con que provilegios (lectura, escritura, etc.)

La información sobre los sistemas de archivos que pueden ser exportados desde una máquina esta contenida en el archivo de configuración /etc/exports.

La sintaxis básica de este archivo es:

```
<sistema de archivos> <regla_IP> (<opciones>), <regla_IP> (<opciones>)
```

Donde <sistema de archivos> es el directorio asociado al sistema de archivos que se quiere exportar, <regla_IP> es una regla que define que Ips pueden montar remotamente con NFS el sistema de archivos y <opciones> son las opciones de NFS que definen los privilegios que se

tienen remotamente sobre el sistema de archivos.

Las opciones que pueden ser ingresadas son:

- ro: directorio de solo lectura
- rw: lectura/escritura
- no_root_squash: permite al usuario root del cliente tener los mismos privilegios administrativos del root en la máquina que exporta el sistema de archivos.
- no_subtree_check: elimina sistema de chequeo que puede hacer más lentas las transferencias de archivos desde el NFS y los sistemas de archivos del cliente.
- sync: las operaciones de escritura en el cliente sobre el NFS se realizan esperando que los archivos sean efectivamente escritos en sistema de archivos real. Impide problemas de corrupcion de datos por interrupciones repentinas de la comunicación.

Un ejemplo de un archivo exports es:

Archivo /etc/exports:

```
/home  * (rw,sync)
/tmp 172.16.2.0/255.255.255.0 (rw,no_root_squash,sync)
/mnt/usbdisk 172.16.0.0/255.255.0.0 (ro,async)
```

NOTA: Para usar el NFS es necesario activar los demonios y habilitar los servicios nfs y nfslock.

Una vez en la lista de FS exportables, se debe proceder a registrar los FS para que efectivamente puedan ser accedidos desde un cliente. Esto se realiza con el comando exportfs.

Comandos 19:

exportfs -av
Exporta todos los directorios definidos en exports

exports -uav
Deshace la exportación de todos los directorios previamente exportados

Una vez los directorios han sido exportados, cualquier máquina que tenga autorizado el montaje puede montar el sistema de archivos remotos usando el comando:

Comandos 20:

mount -t nfs <servidor>:<directorio_exportado> <directorio_cliente>

NOTA: Para realizar esta operacion se recomienda que el cliente tenga activado el demonio y habilitado el servicio nfslock.

2.2.1. Montaje automático de un sistema de archivos NFS

El NFS en nuestro caso sirve para montar el directorio de los usuarios en el servidor como directorio local en los clientes. Así el sistema de autenticación NIS permite el acceso de los usuarios y el NFS les permite tomar control de sus archivos en el servidor.

Hace falta sin embargo configurar un último servicio que permite montar el directorio home solo cuando es necesario evitando de este modo un uso innecesario de la red. El sistema que permite esto se conoce como autofs (Automounter).

Para configurar el autofs en el cliente se debe primero configurar los directorios que serán automaticamente montados. Esto se realiza modificando los archivos de configuración /etc/auto.master y /etc/auto.<directorio> donde <directorio> es el directorio que se automontara.

En el directorio /etc/auto.master se declaran los directorios que se automontaran y el archivo auto asociado a ellos. Así por ejemplo:

Archivo /etc/auto.master:

```
/home /etc/auto.home –timeout 600
```

La opción timeout permite definir un tiempo de 3 minutos para el montaje del directorio. Pasado ese tiempo el sistema de archivos que no sea accedido se desmontará automaticamente.

El archivo auto asociado al directorio de automontaje contiene las opciones de montaje para el sistema de archivos. La sintaxis general del archivo auto es:

```
<directorio> -fstype=nfs,<options> <servidor>:<directorio_remoto>
```

Ejemplo:

Archivo /etc/auto.home:

```
*   -fstype=nfs,soft,intr,rsize=8192,wsize=8192,nosuid,tcp 10.1.1.1:/export/home:&
```

Una vez configurado el sistema de montaje automático del sistema de archivos de usuarios el servicio deberá iniciarse y verificarse:

Comandos 21:

```
# service autofs start
# service autofs status
```

2.3. Autenticación automática de los usuarios

Para conseguir que un usuario se autentique automáticamente en su propia cuenta en el servidor se deben ejecutar la siguiente secuencia de comandos:

Comandos 24:

```
# ssh-keygen -t rsa
(Responder a todo con un <ENTER> (no introducir información))
# cat .ssh/id_rsa.pub >> .ssh/authorized_keys
# chmod og-rw .ssh/authorized_keys
```

Si se desea habilitar la autenticación en una máquina remota en la que no este corriendo un sistema de montaje de sistema de archivos automático se debe ejecutar las siguiente secuencia:

Comandos 25:

```
# ssh-keygen -t rsa
(ejecutar esto solo si no se ha ejecutado previamente el mismo comando)
# scp .ssh/id_rsa.pub <login>@<cliente>:key-servidor.pub
Conectarse al cliente:
# ssh -l <login> <cliente>
Una vez allí ejecutar:
# ssh-keygen -t rsa
# cat key-servidor >> .ssh/authorized_keys
# chmod og-rw .ssh/authorized_keys
```

2.4. Prueba de servicios básicos

Una vez configurados los 3 servicios básicos, a saber el sistema de información por red (NIS), el sistema de archivos por red (NFS) y el servicio de automontaje de los sistemas de archivos del servidor (autofs) es posible ahora hacer una prueba completa de la función de estos servicios.

La prueba se realiza abriendo una sesión en uno de los nodos de un usuario del servidor. El resultado natural de este proceso debe ser el establecimiento de una sesión del sistema operativo sobre el homedirectory de ese usuario. Los comandos para la prueba serían:

Comandos 22:
```
[root@nodo ~]# ssh cluser@localhost
cluser@localhost's password:
Last login: Sun Jul 13 14:42:25 2008 from server.cluster
[cluser@nodo ~]$
```

Parte 3. Servicios especializados de cluster

Existen dos formas de calcular basicas sobre un cluster. La primera y la más sencilla es la de utilizar la disponibilidad de un conjunto de recursos de computo disponibles y de acceso más o menos transparente para los usuarios para ejecutar concurrentemente más de una instancia del mismo programa usando por ejemplo distintos conjuntos de parámetros o distintos datos en cada instancia. El resultado efectivo es el de obtener en un tiempo muy inferior lo que se calcularía con la ejecución sucesiva de cada instancia. A este tipo de aproximación se la conoce como “Paralelismo Embarazoso” (Embarringly Parallelism) y es ampliamente utilizada en la computación de alto rendimiento. El segundo tipo de aproximación es aquel en el que un mismo programa se fragmenta en tiempo de ejecución en “hilos”, instancias paralelas pero relacionadas, que se ejecutan en los distintos procesadores disponibles en el cluster. En este último caso las instancias separadas se comunican entre sí usando diferentes mecanismos. A este tipo de aproximación a los problemas se la llama Paralelismo Puro.

Ambos tipos de aproximaciones requieren diferentes tipos de herramientas. En el primer caso (paralelismo embarazoso) es necesario disponer de un sistema que permita la asignación eficiente de recursos de computo a un conjunto de programas que lo requieran. Este tipo de herramientas se conocen como “programadores de trabajos” o “calendarizadores” (Job Schedulers) y son de las herramientas más utilizadas en la computación en clusters. El paralelismo puro requiere de herramientas normalmente vinculadas con lenguajes de programación (compiladores, librerias,

entre otros)

En ambos casos el almacenamiento masivo de datos es un requerimiento fundamental cuando se trabaja con grandes problemas de computo. La disposición de un sistema que permita reunir las capacidades de almacenamiento de recursos independientes es central en la computación en clusters.

En esta parte mostraremos la manera como pueden instalarse algunas de las más conocidas herramientas para realizar las tareas mencionadas arriba.

3.1. Librerias de computo paralelo

Existen diversos modelos de programación en paralelo, la programación con hilos, la programación con paso de mensajes, la programación con variables compartidas. En esta guía nos concentraremos en las heramientas para programación paralela usando el modelo de paso de mansajes (Message Passing) que por su versatilidad y portabilidad es también el más utilizado en la computación de Alto rendimiento. En particular nos concentraremos en una implementación de la denominada "Message Passing Interface" (interface de paso de mensajes), que se ha convertido hoy por hoy en el estándar de hecho de la programación paralela

3.1.1. MPICH

MPICh o MPI Chamaleon es una de las implementaciones reconocidas del estándar MPI.

La instalación de MPICh en un cluster comienza con la consecución de las fuentes de la librería. Una vez obtenidas las fuentes la instalación sigue el procedimiento estándar de instalación en sistemas *nix (Linux, Unix, BSD):

En el servidor:

<u>Comandos 23:</u>

```
# tar zxvf mpich.tar.gz -C /usr/local/src
```
Desempacar las fuentes en un directorio de manipulación de fuentes

```
# cd /usr/local/src/mpich-1.2.7p1
# RSHCOMMAND=ssh ./configure --prefix=/usr/local
```
Es importante el uso de la variable de ambiente RSHCOMMAND que garantiza que las comunicaciones entre instancias se hagan usando el protocolo SSH

```
# make
```

Si los nodos del cluster son prácticamente idénticos al servidor (misma distribución, igual arquitectura, etc.) los archivos compilados hasta aquí podrían ser copiados a todos los nodos antes del paso final de instalación. Para ello se puede ejecutar, por cada nodo, el commando:

<u>Comandos 24:</u>

```
# cd /usr/local/src
# tar zcvf mpich-bin.tar.gz mpich-1.2.7p1
```

```
# scp mpich-bin.tar.gz nodo1:/usr/local/src
# ssh nodo1 tar zxvf /usr/local/src/mpich-bin.tar.gz -C /usr/local/src
```

Una vez todas las máquinas, servidores y nodos tienen los binarios compilados se procede a instalarlos:

Comandos 25:

```
# cd /usr/local/src/mpich-1.2.7p1
# make install
```

3.1.2. Prueba básica de procesamiento paralelo

Para probar la librería de computo paralelo es necesario abrir una sesión de usuario (en lo sucesivo asumiremos que el usuario es "cluser") en cualquiera de las máquinas del cluster:

Comandos 26:

```
# ssh cluser@server.cluster
```

A continuación es necesario enumerar en un archivo "machines.txt" la lista de los FQHN de las componentes del cluster a las que se tiene acceso y desde las que se podría hacer el lanzamiento de los trabajos. El archivo "machines.txt" debe contener una copia del nombre de cada máquina por cada procesador del que disponga. Así si el servidor tiene 4 procesadores y los nodos 1 y 2 tienen solo 2 el archivo sería:

Archivo machines.txt:

```
server.cluster
server.cluster
server.cluster
server.cluster
nodo1.cluster
nodo1.cluster
nodo2.cluster
nodo2.cluster
```

El orden en el que se consignen las máquinas en el archivo determinará también el orden de utilización de los respectivos procesadores.

Ahora es necesario disponer de un programa de ejemplo para probar la ejecución de programas en paralelo con MPICh. Para ello es posible obtener el programa "cpi.c" disponible en el directorio de fuentes del MPICh:

Comandos 27:

```
$ cp /usr/local/src/mpich-1.2.7p1/examples/basic/cpi.c .
```

El programa debe compilarse usando para ello el script "mpicc" especialmente desarrollado con la interface MPICh:

Comandos 28:

```
$ mpicc cpi.c -o cpi.out
```

NOTA: es importante verificar que el script “mpicc” si corresponda a la versión de MPICh recien instalada. Para ello ejecute:

Comandos 29:

```
$ which mpicc mpirun
/usr/local/bin/mpicc
/usr/local/bin/mpirun
```

Si la salida mostrada no es la que se obtiene tanto mpicc como mpirun (ver abajo) deben ser invocados usando el camino completo de ambos programas (/usr/local/bin/mpicc).

Una vez compilados la ejecución del programa se realiza invocando el script “mpirun” e indicando el número de procesadores a utilizar y el archivo con la lista de procesadores disponibles.   El comando de ejecución es:

Comandos 30:

```
$ mpirun -np 2 -machinefile machines.txt ./cpi.out
Process 0 of 2 on server.cluster
pi is approximately 3.1415926544231318, Error is 0.0000000008333387
wall clock time = 0.003597
Process 1 of 2 on nodo.cluster
```

La salida anterior es correcta indicando que el programa paralelo ha corrido exitosamente usando 2 instancias paralelas.

3.1.3. Prueba avanzada de procesamiento paralelo

Una prueba más compleja de las posibilidades que ofrece el procesamiento paralelo nos la ofrece una poderosa y muy bonita herramienta conocida como POV-Ray. Este programa permite generar imágenes fotorrealistas de escenarios en 3D usando una técnica conocida como “trazado de rayos”. La técnica de trazado de rayos es muy costosa lo que implica que la generación (render) de escenas muy complejas puede llegar a tomar mucho tiempo.   En una de las más reconocidas aplicaciones del cálculo paralelo es posible utilizar una herramienta denominada “MPI POV-Ray” para realizar el rendering de forma más rápida.

Para abreviar usaremos una versión precompilada en arquitectura de 32 bits del programa, mpi-povray31.tar.gz. La instalación de la herramienta es supremamente simple:

Comandos 31:

```
# tar zxvf mpi-povray31.tar.gz  -C /usr/local/src
# cd /usr/local/src/mpi-povray31
# ./install
```

Para realizar la prueba es necesario abrir una sesión como un usuario normal (no root).

Para realizar una prueba de rendering rápida se puede copiar una escena de ejemplo y correrla con una resolución moderada:

Comandos 32:

$ cp -rf /usr/local/src/mpi-povray31/scenes/advanced/woodbox.pov
$ x-povray +FN +I chess2.pov +L/usr/local/src/mpi-povray31/include/ +V +D +W640 +H480

```
...
Displaying...
Using 24 bit TrueColor visual...
  0:02:53 Rendering line  480 of  480.
Done Tracing
chess2.pov Statistics, Resolution 640 x 480
----------------------------------------------------------------------------
Pixels:          307200   Samples:         3407394   Smpls/Pxl: 11.09
Rays:           6843139   Saved:             67409   Max Level: 5/5
----------------------------------------------------------------------------
Ray->Shape Intersection          Tests       Succeeded  Percentage
----------------------------------------------------------------------------
Box                           11404982         2381419     20.88
Cone/Cylinder                 13014936          736743      5.66
CSG Intersection             157682072        23438789     14.86
CSG Union                     72133653        23874491     33.10
Plane                        371402986       216028818     58.17
Quadric                       77743011        23799236     30.61
Sphere                       420650193        40794403      9.70
Bounding Object              343674486        23398358      6.81
----------------------------------------------------------------------------
Calls to Noise:           21880962   Calls to DNoise:       20199136
----------------------------------------------------------------------------
Shadow Ray Tests:         41971212   Succeeded:               417924
Reflected Rays:            3435745
----------------------------------------------------------------------------
Smallest Alloc:                 10 bytes   Largest:            12828
Peak memory used:           699657 bytes
----------------------------------------------------------------------------
Time For Trace:    0 hours  2 minutes  54.0 seconds (174 seconds)
    Total Time:    0 hours  2 minutes  54.0 seconds (174 seconds)
```

Note como el rendering de esta escena con una moderada resolución toma más de 150 segundos en completarse. Si se multiplicara por 4 el tamaño de cada lado, el tiempo de computo se incrementaría en un factor de 16 es decir tomaría cerca de 1 hora.

La misma tarea se puede ejecutar en paralelo usando:

Comandos 33:

$ cp -rf /usr/local/src/mpi-povray31/scenes/advanced/woodbox.pov
$ mpirun -np 2 -machinefile machines /usr/local/bin/mpi-x-povray +FN +I chess2.pov +L/usr/local/src/mpi-povray31/include/ +V +D +W640 +H480

O un detalle de la imágen con:
$ mpirun -np 2 -machinefile machines /usr/local/bin/mpi-x-povray +FN +I chess2.pov +L/usr/local/src/mpi-povray31/include/ +vl +d -x +W640 +H480 +S100 +E150 +SC400 +EC450

3.2. Sistemas de programación de trabajos (job scheduler)

Uno de los sistemas de programación de trabajos más conocidos es el de la SUN, Grid Engine, abreviado SGE.

3.2.1. Instalación y configuración de SGE

La instalación de SGE se realiza siguiendo el siguiente procedimiento:

1. Crear una cuenta y un usuario especial para el sistema

Comandos 34:

```
# useradd -M sgeadmin
```

2. Habilitar el puerto a través del cual se manejan los eventos del sistema. Agregar al archivo /etc/services la línea: `sge_commd 536/tcp # communication port for Grid Engine`

3. Creación del directorio raíz del programa e instalación de los archivos del paquete:

Comandos 35:

```
# mkdir /gridware/sge
# tar -zxvf sge-5.3p6-bin-glinux.tar.gz -C /gridware/sge
# tar -zxvf sge-5.3p6-common.tar.gz -C /gridware/sge
```

4. Configurar variables de ambiente críticas

Comandos 36:

```
# export SGE_ROOT=$SGE_ROOT
# export PATH=$PATH:$SGE_ROOT/bin/glinux
# echo “export SGE_ROOT=$SGE_ROOT” >> /etc/bashrc
# export “PATH=$PATH:$SGE_ROOT/bin/glinux” >> /etc/bashrc
# echo “export SGE_ROOT=$SGE_ROOT” >> /etc/profile
# export “PATH=$PATH:$SGE_ROOT/bin/glinux” >> /etc/profile
```

5. Fijar los permisos de los binarios instalados

Comandos 37:

```
# $SGE_ROOT/util/setfileperm.sh sgeadmin sgeadmin $SGE_ROOT
```

6. Crear un archivo con la lista de todas las máquinas ejecutoras. El archivo puede denominarse “machines-sge.txt” y debería ubicarse en el directoro $SGE_ROOT que en el paso siguiente será copiado a todos los nodos. El archivo machines-sge.txt contiene la lista del nombre de las máquinas. Estos nombres podrán ir o no acompañadas del dominio.

7. Copiado del directorio de instalación a todos los nodos. Una vez el directorio de instalación ha sido configurado apropiadamente se sugiere copiar ese directorio a todos los nodos que correran SGE.

Comandos 38:

```
# cd /
# tar -zcvf /tmp/gridware-bin.tar.gz /gridware
# scp /tmp/gridware-bin.tar.gz nodo:/
```

ssh nodo tar -zxvf /tmp/gridware-bin.tar.gz -C /

8. Instalar en el servidor el servicio “qmaster”

Comandos 39:

```
# cd $SGE_ROOT
# ./install_qmaster
# . $SGE_ROOT/default/common/settings.sh
```

La mayoría de las cuestiones formuladas durante la instalación del qmaster son intuitivas. Algunas importantes cuestiones incluyen: la creación del script de inicialización del demonio (a lo que debe responderse siempre que sí).

9. Instalar en el servidor y los nodos el servicio “execd”

Comandos 40:

```
# cd $SGE_ROOT
# ./install_execd
# . $SGE_ROOT/default/common/settings.sh
```

La mayoría de las cuestiones formuladas durante la instalación del execd son también intuitivas. Algunas importantes cuestiones incluyen: la creación de la cola por defecto para la máquina respectiva (a lo que se debe responder que sí).

3.2.2. Pruebas de SGE

Para probar que el sistema esta en funcionamiento se puede consultar la cola de procesos:

Comandos 41:

```
# qstat -f
queuename            qtype used/tot. load_avg arch       states
----------------------------------------------------------------------------
nodo.q               BIP   0/1       1.03     glinux
----------------------------------------------------------------------------
server.q             BIP   0/1       1.09     glinux
```

Si el ejecutor fue exitosamente instalado en todos los nodos y componentes del sistema la lista de esas componentes debe aparecer al invocar el comando de consulta de colas.

Se puede probar la ejecución de un trabajo de prueba en el sistema de colas con el comando:

Comandos 42:

```
#  qsub $SGE_ROOT/examples/jobs/sleeper.sh
your job 2 ("Sleeper") has been submitted
#  qstat -f
queuename            qtype used/tot. load_avg arch       states
----------------------------------------------------------------------------
nodo.q               BIP   0/1       1.02     glinux
----------------------------------------------------------------------------
server.q             BIP   0/1       1.07     glinux
```

```
############################################################################
 - PENDING JOBS - PENDING JOBS - PENDING JOBS - PENDING JOBS - PENDING JOBS
############################################################################
      2     0 Sleeper    root         qw    07/14/2008 00:38:20
```
qstat -f
```
queuename            qtype used/tot. load_avg arch       states
----------------------------------------------------------------------------
nodo.q               BIP   1/1       1.05     glinux
      2     0 Sleeper    root         r     07/14/2008 00:38:27 MASTER
----------------------------------------------------------------------------
server.q             BIP   0/1       1.07     glinux
```

Para detalles sobre la manera como pueden prepararse y enviarse trabajos al sistema de colas de SGE ver la “Guía Práctica de Linux Clustering con Rocks”.

3.3. Sistemas de almacenamiento distribuido

Uno de los más importantes condicionantes del trabajo en sistemas de computo distribuido como los clusters es la disposición de una capacidad muy grande de almacenamiento. Es normal que los grandes programas de computo utilicen grandes cantidades de datos como entradas o como salidas. Es necesario entonces disponer en el cluster de un sistema que permita el almacenamiento de esas enormes cantidades de datos, que pueden superar la capacidad individual de uno solo de los discos duros involucrados.

3.3.1. PVFS2: Instalación

En el caso de esta guía orientaremos sobre la instalación del PVFS2 (Parallel Virtual File System 2) un sistema de almacenamiento distribuido y en paralelo, liviano, de fácil instalación y uso, aunque probablemente no tan robusto como otros utilizados en ambientes de producción.

La instalación del PVFS2 comienza con la consecución de las fuentes del programa. Una vez conseguidas deben desempacarse en un sitio normalmente preestablecido:

<u>Comandos 43:</u>

```
# tar -zxvf pvfs-2.7.1.tar.gz -C /usr/src
# ln -s /usr/src/pvfs-2.7.1  /usr/src/pvfs2
```

Antes de compilar es necesario verificar primero cuál es la versión del kernel que se esta corriendo y si se dispone de las fuentes de ese mismo kernel. Estas informaciones son de utilidad para fabricar el módulo que permitirá cargar el sistema de archivos distribuido como un sistema de archivos convencional.

<u>Comandos 44:</u>

```
# uname -r
2.6.18-53.el5
Esta es la versión del kernel.  PVFS2 funciona mejor en versiones 2.6.x del kernel

# rpm -qa | grep kernel
kernel-2.6.18-53.el5
```

kernel-headers-2.6.18-53.el5
kernel-devel-2.6.18-53.el5
Así sabremos que paquetes asociados con el kernel están instalados en el sistema.

Para la compilación del módulo que permite el montaje automático del sistema de archivos es necesario que el paquete kernel-devel este instalado. Se puede verificar que las fuentes estén en su lugar con los siguientes comandos:

Comandos 45:

```
# ls -l /usr/src/kernels/
drwxr-xr-x 18 root root 4096 mar  5 10:49 2.6.18-53.el5-i686
```

El directorio que aparece allí contiene las fuentes del kernel requeridas

```
# ls  /lib/modules/2.6.18-53.el5/build
```

Este directorio es un enlace simbólico al directorio con las fuentes del kernel

Una vez verificada la presencia de las fuentes del kernel se procede a configurar y compilar las fuentes del programa:

Comandos 46:

```
# cd /usr/src/pvfs2
# ./configure –with-kernel=/usr/src/kernels/2.6.18-53.el5-i686
# make
```

En este punto se hace necesario compilar también el módulo del kernel que permitirá manipular el sistema de archivos distribuido como un directorio más en el cluster. Para compilar el módulo e instalarlo haga:

Comandos 47:

```
# cd /usr/src/pvfs2
# make kmod
```

Una vez compilado y si las máquinas componentes del cluster son similares es posible distribuir el paquete compilado por todos los nodos así:

Comandos 48:

```
# cd /usr/src
# tar -zmcvf pvfs2-binaries.tar.gz pvfs-2.7.1 pvfs2
# scp pvfs2-binaries.tar.gz nodo.cluster:/usr/src
# ssh nodo.cluster tar -zxvf /usr/src/pvfs2-binaries.tar.gz -C /usr/src
```

Una vez los binarios hayan sido compilados y estén presentes en todas las máquinas del cluster ahora es posible correr la tarea de instalación. En el servidor y en cada nodo ejecutar:

Comandos 49:

```
# cd /usr/src/pvfs2
# make install
# make kmod_install
```

Ahora estamos listos para la configuración del sistema.

3.3.2. PVFS2: Configuración

La configuración del sistema implica primero decidir cuáles serán los roles de cada componente (servidor y nodos) en el cluster. Existen 3 roles básicos:

- Servidor de metadatos (metadata server): estas son componentes fundamentales del sistema. Los servidores de metadatos almacenan la información sobre los archivos en el sistema. Cada vez que se quiere recuperar un objeto los clientes deben consultar a un servidor de metadatos para que le indique la información sobre el objeto.

- Servidor de entrada/salida (I/O server): estas son las componentes que efectivamente almacenan los datos en el sistema. Normalmente los datos se fragmentan entre los I/O server y algunos paquetes duplicados se almacenan en múltiples I/O server para asegurar la tolerancia a las fallas del sistema.

- Cliente. Esta es una máquina cualquiera en el sistema que esta configurada para acceder a los datos almacenados en el sistema de archivos distribuido.

En principio todas las máquinas de un cluster jugar todos estos roles. Sin embargo por comodidad y seguridad escogeremos que todos los nodos tendrán el rol de I/O servers y solo el servidor se comportará como un metadata server.

Una vez decidida la distribución de roles se debe preparar el archivo de configuración básica del sistema. Esto se consigue con el comando:

<u>Comandos 50:</u>

```
   # pvfs2-genconfig  /etc/pvfs2-fs.conf
**********************************************************************
    Welcome to the PVFS2 Configuration Generator:

This interactive script will generate configuration files suitable
for use with a new PVFS2 file system.  Please see the PVFS2 quickstart
guide for details.

**********************************************************************
You must first select the network protocol that your file system will use.
The only currently supported options are "tcp", "gm", "mx", "ib", and "portals".
(For multi-homed configurations, use e.g. "ib,tcp".)

* Enter protocol type [Default is tcp]:

Choose a TCP/IP port for the servers to listen on.  Note that this
script assumes that all servers will use the same port number.

* Enter port number [Default is 3334]:

Choose a directory for each server to store data in.
```

```
* Enter directory name: [Default is /pvfs2-storage-space]:

Choose a file for each server to write log messages to.

* Enter log file location [Default is /tmp/pvfs2-server.log]: /var/log/pvfs2-server.log

Next you must list the hostnames of the machines that will act as
I/O servers.  Acceptable syntax is "node1, node2, ..." or "node{#-#,#,#}".

* Enter hostnames [Default is localhost]: server nodo

Now list the hostnames of the machines that will act as Metadata
servers.  This list may or may not overlap with the I/O server list.

* Enter hostnames [Default is localhost]: server

Configured a total of 2 servers:
1 of them are I/O servers.
1 of them are Metadata servers.

* Would you like to verify server list (y/n) [Default is n]? y

****** I/O servers:
server nodo

****** Metadata servers:
server

* Does this look ok (y/n) [Default is y]?

Writing fs config file... done
```

Nótese que el valor por defecto del archivo para la bitácora del sistema ha sido cambiado respecto a su valor por defecto.

Una vez generado el archivo de configuración es necesario que este archivo este presente en todos los nodos del cluster.

Finalmente para comenzar a utilizar el sistema de archivos distribuidos es necesario ejecutar en cada “servidor” pvfs2 el demonio correspondiente al servicio. Los comandos requeridos son:

<u>Comandos 51:</u>

pvfs2-server /etc/pvfs2-fs.conf -f
Esta primera corrida permite crear (si no se ha hecho) el archivo de almacenamiento de los archivos asociados al pvfs2. Por defecto el nombre del directorio es /pvfs2-storage-space

pvfs2-server /etc/pvfs2-fs.conf
Este es el comando que finalmente pone a funcionar los servidores

Una vez los servidores están corriendo es posible configurar y poner en funcionamiento al menos un cliente.

La configuración del cliente comienza creando el directorio particular que estará asociado, como es común en Linux, al sistema de archivos distribuidos de pvfs2. Normalmente se elige

/mnt/pvfs2.

Comandos 52:

```
# mkdir -p /mnt/pvfs2
```

Para indicar la manera como el sistema de archivos de pvfs2 se vincula con el directorio /mnt/pvfs2 se debe crear el archivo /etc/pvfs2tab, de formato análogo al del /etc/fstab:

Archivo machines.txt:

```
tcp://nodo:3334/pvfs2-fs /mnt/pvfs2 pvfs2 defaults,noauto 0 0
```

Una vez definida la manera de acceder al sistema de archivos de pvfs2 es posible hacer una prueba de conectividad al sistema:

Comandos 53:

```
# pvfs2-ping -m /mnt/pvfs2
(1) Parsing tab file...

(2) Initializing system interface...

(3) Initializing each file system found in tab file: /etc/pvfs2tab...

   PVFS2 servers: tcp://nodo:3334
   Storage name: pvfs2-fs
   Local mount point: /mnt/pvfs2
   /mnt/pvfs2: Ok

(4) Searching for /mnt/pvfs2/ in pvfstab...

   PVFS2 servers: tcp://nodo:3334
   Storage name: pvfs2-fs
   Local mount point: /mnt/pvfs2

   meta servers:
   tcp://server:3334

   data servers:
   tcp://server:3334
   tcp://nodo:3334

(5) Verifying that all servers are responding...

   meta servers:
   tcp://server:3334 Ok

   data servers:
   tcp://server:3334 Ok
   tcp://nodo:3334 Ok

(6) Verifying that fsid 2016931876 is acceptable to all servers...

   Ok; all servers understand fs_id 2016931876

(7) Verifying that root handle is owned by one server...

   Root handle: 1048576
     Ok; root handle is owned by exactly one server.

=============================================================

The PVFS2 filesystem at /mnt/pvfs2/ appears to be correctly configured.
```

En la sálida anterior todo parece estar en orden. Existe una familia completa de comandos "pvfs2-" que permiten manipular de una manera sencilla el sistema de archivos. Así por ejemplo

“pvfs2-ls” permite revisar el contenido del sistema de archivos, “pvfs2-cp” permite copiar archivos hacia y desde el sisyema de archivos y así sucesivamente.

La manera más normal y natural de interactuar con el sistema de archivos es a través del módulo del kernel que compilamos para ese fin. Para utilizar el módulo se debe completar el siguiente procedimiento:

Comandos 54:

insmod /usr/src/pvfs2/src/kernel/linux-2.6/pvfs2.ko
Insertar el módulo del kernel

cd /usr/src/pvfs2/src/apps/kernel/linux
pvfs2-client starting

```
Spawning new child process
Waiting on child with pid 3215
About to exec: ./pvfs2-client-core, with args: pvfs2-client-core -a 5 -n 5 --logtype
file -L /tmp/pvfs2-client.log
```

Una vez habilitado el cliente basta con montar finalmente el sistema de archivos en el directorio creado para ese fin:

Comandos 55:

mount -t pvfs2 tcp://nodo:3334/pvfs2-fs /mnt/pvfs2
df -h /mnt/pvfs2

```
S.ficheros          Tamaño Usado  Disp Uso% Montado en
tcp://nodo:3334/pvfs2-fs
                       21G  8,2G   13G  40% /mnt/pvfs2
```

Una vez como directorio la manipulación del sistema de archivos se vuelve prácticamente trivial.

Parte 4. Procedimientos básicos

4.1. Creación de cuentas de usuario en linux.

Para crear cuentas de usuario en Linux es necesario primero disponer de un directorio para los directorios 'casa' (home directory) de los usuarios. En un sistema de computación en Grid lo mejor es disponer de un directorio distinto a aquel que se usa por defecto en linux (/home). Se puede crear el directorio **/export/home** en su lugar así:

Comandos 22:

mkdir -p /export/home

Una vez disponible este directorio la creación de una cuenta de usuario procede de la siguiente manera:

Comandos 23:

useradd <login> -d /export/home/<login> -m

donde <login> es el nombre de usuario válido en linux.
El password del usuario se puede fijar usando el comando:

Comandos 24:

```
# passwd <login>
```